A VEHICLE ASSIGNMENT PROBLEM ALGORITHM

A THESIS

Presented to

The Faculty of the Graduate Division

by

John Edgar Felch, Jr.

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Operations Research

Georgia Institute of Technology

June, 1974

A VEHICLE ASSIGNMENT PROBLEM

ALGORITHM

Approved:

C. M. Shetty, Chairman

Richard H. Deane

Donovan B. Young

Date approved by Chairman: May 16, 1974

ACKNOWLEDGMENTS

The contribution of Professor C. M. Shetty to the completion of this paper cannot be easily expressed. His keen insight and patience during long discussions is greatly appreciated. His guidance, instruction, and inspiration will remain a most significant part of my academic experience.

I would also like to thank the members of the reading committee, Dr. D. B. Young and Dr. R. H. Deane, for their willingness to give of their valuable time in reviewing and evaluating this work. The programming assistance of Dr. R. L. Rardin is also acknowledged with grateful appreciation.

Finally, I would like to formally thank my wife, Judy, for her encouragement, understanding, and faith. Her selfless typing of the drafts of this thesis made the process much smoother than would have otherwise been possible. And most important, her management of everything else allowed me to devote my time to this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

SUMMARY

This study deals with the problem of assigning vehicles, available at several depots, to meet the requirement for vehicles at several job sites. A vehicle may perform more than one job after it leaves the depot, but must return to the originating depot. Each job requirement essentially states the number and type of vehicle required, the time required, and the release time.

The model for the above problem involves a large number of variables and equations and would be beyond the capacity of present computers. The strategy used is first to identify null variables based on the dynamic constraints of the problem. Then the resulting static problem is solved by Dantzig-Wolfe decomposition where each subproblem turns out to be a shortest path problem for which efficient network codes are available. Finally, the resulting solution is integerized if not already so. This last phase is heuristic but easily implemented.

The algorithm is applied to some test problems and the results are encouraging. Larger problems are not solved because of memory storage restrictions. No attempt has been made to optimize the program. However, the study does discuss the size of problems that one can expect to handle if a more efficient means of storing information is adopted.

CHAPTER I

INTRODUCTION

Although operations research has made significant contributions to all phases of the transportation industry over the last twenty years, we shall deal in this paper only with the particular aspects of the transportation field associated with vehicle fleet management. The problems considered in this study are of interest both academically and commercially. Academically, the problem is related to other problems in scheduling, network theory, and assembly-line balancing. Obvious commercial value is achieved by any successful effort which results in cost reduction or improved efficiency.

Let us consider the problem of a large company which provides a fleet of trucks (or other vehicles) for use by numerous clients on a daily basis. The company must make long-range decisions to determine how large a fleet to maintain and where to locate it, which will require the establishment of one or more depot sites. Given these, the company must determine, on a periodic basis, the most efficient means of assigning its fleet to its clients depending upon the demand and limitations (constraints) imposed on the dispatcher. This is a combinatorial problem and is quite complex to solve. Besides, the complexity grows exponentially with problem size. It is this type of problem that we are addressing in this study.

Operations research literature refers to all three of the pre-

dominant problems of fleet management mentioned above, i.e., depot siting, vehicle assignment, and fleet size determination. The initial operational problem of a new firm would be the selection of an optional number of vehicle depots at the best set of locations or sites. A similar decision process is required when an established firm desires to add a new depot or consolidate existing ones. A detailed discussion of this problem is given by Eilon, et al. [11], so it will not be further considered here. Similarly, fleet size determination is beyond the scope of this research. Once the appropriate vehicle assignment has been established, algorithms are available for generating the minimum fleet size required to satisfy the fixed schedule. Fleet size determination problems are discussed in several places [9, 16, and 27]. This research will address the problem of vehicle assignment.

Vehicle assignment is one of the common problems of operating transportation companies, and is the subject of this study. For the sake of clarity we will elaborate on certain key terms used in the study. First, the vehicle is that entity which is dispatched to satisfy the requirements of a client. A depot is some location where we can store and maintain our vehicles when they are not in use. And finally, a job is the location where the vehicle is utilized by a client, without any restrictions or controls imposed by the fleet owners. The environment with which we shall be dealing consists of a fleet manager who must allocate his vehicles to the jobs. He could choose to meet all of the job demands directly from the nearest

depot and return the vehicle directly to that depot upon completion of the job. This does not seem very efficient since some vehicles may be able to complete one job and proceed directly to another with some cost savings over the return-to-depot plan.

The motivation for this research comes from the case study by Gavish and Schweitzer [14] of a trucking company which had a vehicle assignment problem. The algorithm set forth in this paper is capable of handling a more complex and more realistic version of the problem addressed by them.

Literature Review

Throughout the literature on vehicle assignment problems various descriptors are used to title the vehicle assignment problem. These include vehicle dispatching, vehicle scheduling, vehicle assignment, delivery, transportation, and sequencing. There is no significant difference in meaning intended by these different titles. The primary distinction in vehicle assignment problems is between those that permit only one depot to be considered and those that consider several depots. We shall refer to these respectively as central-depot and multiple-depot problems. Secondary classification of the literature can be accomplished in terms of dynamic vs. static systems and stochastic vs. deterministic demands. The majority of the literature was found to be static and deterministic. Only Tillman [37] considers the stochastic demand conditions and only Gavish and Schweitzer [14] postulate a truly dynamic model.

Dantzig was the earliest referenced author to formulate what

we will term the central-depot vehicle assignment problem. It is interesting to trace his work in this area. First [8] he addressed a typical transportation problem of shipping a product from m sources to n destinations. Then Dantzig and Fulkerson [9] considered a fleet size determination problem, and in 1959 Dantzig and Ramser [10] formulated the first classical central-depot vehicle assignment problem.

Minor variations of this problem have been considered, resulting in additional solution difficulties. These variations are as numerous as the articles in this field and are not of sufficient importance to review in detail. The core central-depot problem can be stated as follows:

> N customers with known locations and demands are to be supplied from a single depot by vehicles of known capacity. The problem is to select the best routes subject to:
>
> (i) meeting all demands
> (ii) not exceeding vehicle capacity
> (iii) a maximum time limit for each route
> (iv) a time interval within which a particular demand must be met

From 1962 to 1972, several authors presented solution procedures for the central-depot vehicle assignment problem and its variations. The resultant solution methodology was diverse in that some methods yielded optimal results and some only near optimal, and in that some methods were highly analyical and some highly heuristic. We find that simulation was used in three papers [4, 30, and 34]; integer programming in only one [1]; branch and bound techniques in four [5, 11, 18, and 33]; marginal analysis in five [5, 6, 13, 23,

and 38]; the 3-optimal tour method in two [5 and 11]; linear programming in two [8 and 21]; heuristic programming in nine [6, 10, 13, 17, 18, 23, 28, 38, and 43]; dynamic programming in one [19]; network techniques in two [2 and 20]; and one independent algorithm [22].

The central-depot problem is a special case of the multiple-depot problem, but the solution techniques of the central-depot problems do not extend themselves to the more general problem. For this reason we will not elaborate on the problem statements and solution techniques found in the literature on the central-depot problem.

The next logical advance in the state of the art was the formulation and solution of the multiple-depot vehicle assignment problem. Szwarc was the first author to formulate this problem in 1967. His problem [36] assumes the vehicles return to a depot after completing a job. Szwarc's solution procedure was developed in a transportation network format.

Some logical extensions of Szwarc's work would be the addition of a trans-shipment capability and provision for using arrival intervals for the job demands. These extensions were made as summarized in Table 2, but various solution techniques were used and each had its own peculiarities. Two heuristic programming solutions to the multiple-depot vehicle assignment problem were found [37 and 47]. Neither of these is able to guarantee that an optimal solution will be found, and both neglect the time interval restriction for meeting the demands. Two of the most recent treatments of this problem [3 and 32] also neglect the time interval restriction and the requirement for a vehicle to return to its depot of origin upon completion

of its route. This latter constraint was satisfied by the previously mentioned formulations. Additionally, a linear programming approach [32] ignores the trans-shipments between jobs which are handled by [3] an out-of-kilter network solution. In each of the above cases the problem was defined so as to suit a particular technique. The solution procedures can not be extended to the more general problem we are considering and therefore, we will not dwell on any of these problem statements or solution techniques.

The primary reference for this research effort was a paper by Gavish and Schweitzer [14]. Their formulation neglects the requirement that a vehicle return to its original depot, but they are the first authors to consider the time interval constraints in a multiple-depot vehicle assignment context. Their solution methodology was of the transportation network form. Another consideration introduced to this problem by them is the inclusion of operational (dynamic) constraints on the compatibility of two jobs. This is easily handled by defining a feasible set of job combinations.

To summarize, the major constraints considered in the multiple-depot vehicle assignment problem are:

(i) meet all demands from the depots,
(ii) do not exceed vehicle capacity,
(iii) do not exceed an implicit maximum time limit for each route,
(iv) perform each demand within a specified time interval,
(v) allow for the serial combination of jobs, and
(vi) take into consideration compatibility constraints.

The present research has incorporated all of the constraints

which other multiple-depot problems have considered, with the exception of the re-dispatch provision. In addition, we have added the constraint that each vehicle must return to the depot of origin. Thus, this model is a more versatile one. The decomposition approach we use as part of our procedure has helped substantially to reduce computation time. Our solution technique will provide a near optimal solution with reasonable efficiency.

An overall appreciation of the literature can be obtained by the analysis of Tables 1 and 2. Table 1 presents a classification of articles by problem area with a detailed sub-division of the central-depot assignment problem solution techniques. At a glance one can see the time frame and quantity of work done in a particular area. The vehicle assignment literature cited is comprehensive, however the depot siting and fleet size references are not. These have been included only to provide a broader view of the environment of vehicle assignment problems. Much additional literature is currently available in these two areas. Table 2 provides a detailed analysis of the multiple-depot assignment problem statements and solution techniques used in the referenced literature and this paper.

The intent of this research is to formulate a multiple-depot vehicle assignment problem which will incorporate the key aspects mentioned above that have been omitted from previous work. With the exception of the re-dispatch provision, a general formulation of the problem will be solved by a combination of linear programming and network techniques and a near optimal solution achieved.

Table 1. Literature Synopsis

Part 1.

Reference	8	9	10	19	1	6	20	4	13	17	18	21	23	34	36	2
Year Published	51	54	59	62	64	64	64	67	67	67	67	67	67	67	67	68
Depot Siting																
Fleet Size		+														
Multiple-Depot															+	
Central-Depot																
Linear Prog.	+											+				
Heuristic Prog.			+			+			+	+	+		+			
Dynamic Prog.				+												
Integer Prog.					+											
Marginal Anal.						+			+				+			
Network Tech.							+									+
Simulation								+						+		
Branch & Bound											+					

Part 2.

Reference	22	38	43	5	16	33	37	27	11	39	3	28	32	47	14	30
Year Published	68	68	68	69	69	69	69	70	71	71	72	72	72	72	73	UNK
Depot Siting									+							
Fleet Size					+			+								
Multiple-Depot							+			+	+		+	+	+	
Central-Depot																
Other	+															
Marginal Anal.		+		+												
Heuristic Prog.		+	+									+				
Branch & Bound				+		+			+							
3-Optimal Tour				+					+							
Simulation																+

Table 2. Comparison of Multiple-Depot Formulations

Reference	36	37	39	3	32	47	14	Present Research
Year Published	67	69	71	72	72	72	73	74
Problem Criteria								
Multiple-Depots	yes	yes	yes	yes	yes	yes	yes	yes
Multiple Demand Points	yes	yes	yes	yes	yes	yes	yes	yes
Transhipment allowed	NO	yes	yes	yes	NO	yes	yes	yes
Return to Origin	yes	yes	yes	NO	NO	yes	NO	yes
Cost Minimization	yes	yes	yes	yes	yes	yes	yes	yes
Meet all Demands	yes	yes	yes	yes	yes	yes	yes	yes
Arrival Interval	NO	NO	NO	NO	NO	NO	yes	yes
Re-dispatch Allowed	yes	NO	NO	yes	yes	NO	NO	NO
Compatability Constraints	NO	NO	NO	NO	NO	NO	yes	yes
Number Demands per Day	68			136		183	100	100
Solution Techniques								
Linear Programming					+			+
Heuristic Programming		+	+			+		
Network Techniques	+			+			+	+

Problem Statement and Mathematical Model

To discuss the problem addressed in this study in detail, consider a fixed number of a particular type vehicle assigned to each of several depots. This will specify the number of depots and the quantity of vehicles available at each depot. Now a group of, say, J clients will make known their demands. Each demand will include the number of vehicles required, the type and quantity of cargo to be carried, any special features required on the vehicles, the starting time and place and the job duration or a specified release time and place. The fleet manager must now determine the costs and times for traveling to, from, and between the job locations. He must also specify a safety margin to be added to the time for each route and the maximum time a vehicle will be allowed to wait if it arrives prior to the job starting time, so that the drivers will not be allowed excessive idle time. Typically, there may be 50 to 100 demands per day, as in the case study [14] upon which this paper is based.

The solution of this problem involves the serial combination of jobs subject to the following nine constraints.

(i) The number of vehicles departing or returning to a depot cannot exceed the number assigned to that depot.
(ii) Vehicles must return to their depot of origin.
(iii) A vehicle may be dispatched only once. (By checking to see if an early returning vehicle can accept the entire route of a late starting vehicle, this restriction can be relaxed.)
(iv) The number of vehicles of a particular type departing a job must equal the number of that type that arrived at the job.
(v) The number of vehicles arriving at a job must equal the number requested.
(vi) Vehicles cannot arrive late for a job.
(vii) Vehicles cannot arrive too early for a job.

(viii) Successive cargos must be compatible
(ix) Successive special feature requirements must be compatible.

The objective function will be defined by a cost function which represents the cost of empty vehicles traveling the appropriate routes. This problem will only implicitly consider delivery aspects such as vehicle capacity. We will assume that the activity at a demand location is determined by the user and is therefore an uncontrolled factor for the fleet manager. Therefore, minimizing the costs associated with deadheading of the empty vehicles will minimize the controllable operating costs of the fleet.

In order to develop a solution technique for this multiple-depot vehicle assignment problem, we need to develop the mathematical model for the problem. A summary of the notation is presented in Table 3 for reference throughout the remainder of this paper.

Let us consider a situation with D depots, each having S_i, $i = 1, \ldots, D$ vehicles available for dispatch to J job sites, $D + 1$ to $D + J$, with demand requirements M_j, $j = D + 1, \ldots, D + J$. For convenience in notation we will let $I = D + 1$ and $K = D + J$. Additionally, let us adopt the superscript, k, where $k = 1, \ldots, D$ to represent the depot from which a particular set of vehicles originated. We shall refer to this as the kth type of vehicle but keep in mind that it only identifies the source of the vehicle and does not imply any physical differences between the vehicles.

Our decision variables will be X_{ij}^k which will represent the number of vehicles, originating from the kth depot, to be sent from

Table 3. Notation

Notation	Definition
C^{o}_{ij}	The cost of traveling from depot i to job origin j where i = 1, ..., D and j = I, ..., K
C_{ij}	The cost of traveling from job i to job j where i = I, ..., K and j = I, ..., K
C^{r}_{ji}	The cost of traveling from job release j to depot i where j = I, ..., K and i = 1, ..., D
M_j	The number of vehicles required at job j where j = I, ..., K
S^{k}_{i}	The number of vehicles of type k assigned to depot i where k = 1, ..., D and i = 1, ..., D ($S^{k}_{i} = S^{i}_{i}$ if k = i, 0 otherwise)
J	The total number of job demands
D	The total number of depots, also vehicle types
I	Equals D plus 1
K	Equals D plus J
X^{k}_{ij}	The number of vehicles of type k to be sent from location i to j (the decision variable) where k = 1, ..., D, i = 1, ..., K, and j = 1, ..., K
Ω	The set of null variables
t^{o}_{j}	The origination time for job location j where j = I, ..., K
t^{r}_{j}	The release time for job location j where j = I, ..., K
t_{ij}	The travel time from job location i to j where i = I, ..., K and j = I, ..., K
n_{ij}	The time safety margin for travel from job location i to j where i = I, ..., K and j = I, ..., K
w_j	The maximum waiting time allowed prior to t^{o}_{j} at job location j where j = I, ..., K
g_{ij}	The special feature compatibility index from job location i to j (g_{ij} = 1 if cargos are compatible, 0 otherwise) where i = I, ..., K and j = I, ..., K
f_{ij}	The special feature compatibility index from job location i to j (f_{ij} = 1 if features are compatible, 0 otherwise) where i = I, ..., K and j = I, ..., K

location i to location j. Thus we need to minimize the costs of sending empty vehicles of all types from any depot to any job plus the cost of trans-shipping empty vehicles from any job to any other job and finally the costs of returning empty vehicles of the kth type to the correct originating depot from any job site. These three costs can be respectively expressed mathematically as our objective function:

$$\text{Min } f(X) = \sum_{i=1}^{D} \sum_{j=I}^{K} \sum_{k=1}^{D} C_{ij}^{o} X_{ij}^{k} + \sum_{i=I}^{K} \sum_{j=I}^{K} \sum_{k=1}^{D} C_{ij} X_{ij}^{k} + \sum_{j=I}^{K} \sum_{i=1}^{D} \sum_{k=1}^{D} C_{ji}^{r} X_{ji}^{k} \qquad (1)$$

where C_{ij}^{o} is the cost of sending one vehicle from depot i to job origin j; C_{ij} is the cost of sending one vehicle from job i to job j; and C_{ji}^{r} is the cost of returning one vehicle from job j to depot i.

Now that we have derived our objective function, we must turn our attention to the constraints of the problem. First we must prohibit the dispatch of more vehicles than are available at a particular depot. Clearly there will be no vehicles at depot i of any type other than k = i, since k is an index of the depot of origin. So let us define the vehicles of type k available at depot i to be S_i^k, where $S_i^k = 0$ if $k \neq i$. Thus our constraint for the limitation on supply can be written as:

$$\sum_{j=I}^{K} X_{ij}^{k} \leq S_i^k \quad \text{for all } i = 1, \ldots, D \text{ and } k = 1, \ldots, D \qquad (2)$$

Furthermore, we must be able to guarantee that all of the vehicles that leave a given depot return to that depot at the end of the day.

$$\sum_{j=I}^{K} X_{ji}^{k} = \sum_{j=I}^{K} X_{ij}^{k} \quad \text{for all } i = 1, \ldots, D \text{ and } k = 1, \ldots, D \qquad (3)$$

Equation (2) and (3) are the mathematical representation of conditions (i), (ii), and (iii) as they were described in our problem statement earlier. These are the constraints on the depot environment of our problem and in network theory terminology are the node balance equations for the depots. Now we must examine the job environment and derive the job node balance equations.

First, we will consider the problem of balancing the arrival and departure of the kth type vehicle at job j. These vehicles can arrive (in general) from any depot or be trans-shipped from some other job. Similarly, these vehicles will be sent out from job j to depots or other jobs. Hence we require four terms to express the overall balance of the kth type vehicle at job j.

$$\sum_{i=1}^{D} X_{ij}^{k} + \sum_{i=I}^{K} X_{ij}^{k} - \sum_{i=1}^{D} X_{ji}^{k} - \sum_{i=I}^{K} X_{ji}^{k} = 0 \qquad (4)$$

for all $j = I, \ldots, K$ and $k = 1, \ldots, D$

Also we must insure that the demand of each job site is satisfied, since we assumed that we had the capability to accomplish this and we do not want to disappoint our customers. Here we must consider the inflow of vehicles of all types from all depots and all other

jobs which must equal the number of vehicles demanded, i.e.

$$\sum_{i=1}^{D}\sum_{k=1}^{D} X_{ij}^{k} + \sum_{i=I}^{K}\sum_{k=1}^{D} X_{ij}^{k} = M_j \quad \text{for all } j = I, \ldots, K \tag{5}$$

Equations (4) and (5) are the mathematical formulation of conditions (iv) and (v), respectively of the problem statement. We also need to apply a nonnegativity constraint on our decision variables and require them to be integer valued since we are dealing with whole vehicles.

$$\text{all } X_{ij}^{k} \geq 0, \text{ integer} \tag{6}$$

Equation (1) subject to the constraints of equations (2) through (6) represents the principle expression of the general problem. Our solution technique will focus on these equations, which describe the static aspects of our stated problem.

Following Gavish and Schweitzer [14] we will consider the dynamic aspects of the problem separately.

As defined in Table 3, we will set $X_{ij}^{k} = 0$ if $X_{ij}^{k} \in \Omega$ i.e., the set Ω defines the set of <u>null variables</u>. Now

(a) $X_{ij}^{k} \in \Omega$ if $t_i^r + t_{ij} + h_{ij} \geq t_j^o$ i.e., if the release time at node i plus the travel and safety time between i and j exceed the starting time at node j, then X_{ij}^{k} is a null variable.

(b) $X_{ij}^k \in \Omega$ if $t_i^r + t_{ij} \leq t_j^o - w_j$ i.e., if the earliest (release time plus only travel time) the vehicle can arrive from node i at node j is more than the prescribed waiting time before the start of job j, then X_{ij}^k is a null variable.

(c) $X_{ij}^k \in \Omega$ if $g_{ij} = 0$ i.e., if jobs i and j are not compatible by the nature of their cargo.

(d) $X_{ij}^k \in \Omega$ if $f_{ij} = 0$ i.e., if jobs i and j are not compatible due to special requirements.

The above specifications correspond to conditions (vi) through (ix) of the problem statement. For $X_{ij}^k \in \Omega$, we force $X_{ij}^k = 0$ by setting $C_{ij}^k = \infty$.

There is one other major simplification which can be made to reduce the complexity of our problem expression.

Lemma 1. For each i = 1, ..., D and j = I, ..., K, $X_{ij}^k = X_{ji}^k = 0$ when $k \neq i$ in any feasible solution to the problem.

Proof. There are no vehicles from other depots allowed at depot i, hence, the definition of $S_i^k = 0$ for $i \neq k$. Now from equation (2) we can see that $\sum_{j=I}^{K} X_{ij}^k \leq S_i^k = 0$ for $i \neq k$ and we know that the sum of nonnegative (from equation (6)) variables must be nonnegative and we get: $0 \geq \sum_{j=I}^{K} X_{ij}^k \leq 0$ for $i \neq k$, hence $\sum_{j=I}^{K} X_{ij}^k$, $i \neq k$ must equal zero. Then for $i \neq k$ we get equation (3) in the form $\sum_{j=I}^{K} X_{ji}^k = \sum_{j=I}^{K} X_{ij}^k = 0$ for $i \neq k$. We know that the only way a sum of nonnegative variables can be equal to zero is for all the variables to be individually equal to zero. Hence: $X_{ij}^k = X_{ji}^k = 0$

when $i \neq k$.

By invoking the results of Lemma 1 we can eliminate several summations and match the k-type superscript with the i-depot subscript in certain of our equations. The problem can be restated as:

Problem P: Minimize

$$f(X) = \sum_{k=1}^{D} \sum_{j=I}^{K} C_{kj}^{o} X_{kj}^{k} + \sum_{i=I}^{K} \sum_{j=I}^{K} \sum_{k=1}^{D} C_{ij} X_{ij}^{k} + \sum_{j=I}^{K} \sum_{k=1}^{D} C_{jk}^{r} X_{jk}^{k} \quad (7)$$

Subject to:

$$\sum_{j=I}^{K} X_{kj}^{k} \leq S_{k}^{k} \quad \text{for all } k = 1, \ldots, D \quad (8)$$

$$\sum_{j=I}^{K} X_{jk}^{k} - \sum_{j=I}^{K} X_{kj}^{k} = 0 \quad (9)$$

for all $k = 1, \ldots, D$

$$X_{kj}^{k} + \sum_{i=I}^{K} X_{ij}^{k} - X_{jk}^{k} - \sum_{i=I}^{K} X_{ji}^{k} = 0 \quad (10)$$

for all $k = 1, \ldots, D$ and $j = I, \ldots, K$

$$\sum_{k=1}^{D} X_{kj}^{k} + \sum_{i=I}^{K} \sum_{k=1}^{D} X_{ij}^{k} = M_{j} \quad (11)$$

for all $j = I, \ldots, K$

and all $X_{ij}^{k} \geq 0$, integer (12)

where $C_{ij} = \infty$ if $X_{ij}^{k} \in \Omega$ for any $k = 1, \ldots, D$ and any job combination (i,j).

This formulation is clearly an integer programming problem but its size could still pose a significant computational difficulty even

if the integrality requirement is relaxed. The context of our problem is a daily determination of vehicle assignments for up to ten depots and up to 100 job demands. A simplex solution of this size problem would require 1,002,000 real variables plus 1120 slack and artificial variables and there would be 1120 constraint equations. Thus even a moderate size problem will require some ingenuity and innovation to make a linear programming solution technique feasible. The problem will lend itself well to three different solution techniques, as evidenced by the work of previous authors summarized in Table 2; however, this research will show that employing the Dantzig-Wolfe decomposition principle together with the use of a simple algorithm from network theory (namely, the shortest path algorithm) will yield an efficient algorithm.

CHAPTER II

SOLUTION PROCEDURE

The Algorithm

To solve the problem we will first identify the null variables by the four dynamic considerations discussed earlier. This leaves us with an integer programming problem. We will relax the integrality requirement and solve the resulting linear program by the Dantzig-Wolfe decomposition procedure. The subproblems are selected so that they are shortest-path problems for which efficient procedures for solving are available. The subproblem may have negative cycles and hence, the Dantzig, Blattner and Rao algorithm [49] is used. The specific details of the Dantzig-Wolfe procedure and the shortest-path algorithm are well known and are not discussed here. The solution thus obtained is modified to give an integer solution to problem P. A heuristic procedure is described for achieving this.

To solve the linear program, recall that the general matrix formulation of a decomposable problem is:

$$\text{Min} \quad f(X) = \sum_{k=1}^{D} C_k X_k \tag{13}$$

$$\text{subject to:} \quad \sum_{k=1}^{D} A_k X_k = b_o \tag{14}$$

$$B_k X_K = b_k \quad \text{for all } k = 1, \ldots, D \tag{15}$$

$$\text{all } X_k \geq 0 \tag{16}$$

where D is the number of subproblems.

This is the form of our problem with equation (7) corresponding to (13), equations (8), (9), and (10) corresponding to the subproblem equation (15) for each k, and equation (11) corresponding to the coupling constraint of equation (14). In writing a detailed expression of the kth Dantzig-Wolfe subproblem we can omit the use of the k superscript since it is the same within each subproblem. We will also denote the modified cost coefficients by $\hat{C}^{o}_{kj}$, $\hat{C}_{ij}$, and $\hat{C}^{r}_{jk}$ as obtained by the Dantzig-Wolfe decomposition principle. Thus we get:

Problem P_k:

$$\text{Min } f_k(X) = \sum_{j=I}^{K} \hat{C}^{o}_{kj} X_{kj} + \sum_{i=I}^{K} \sum_{j=I}^{K} \hat{C}_{ij} X_{ij} + \sum_{j=I}^{K} \hat{C}^{r}_{jk} X_{jk} \quad (17)$$

$$\text{subject to: } \sum_{j=I}^{K} X_{kj} \leq S_k \quad (18)$$

$$\sum_{j=I}^{K} X_{jk} - \sum_{j=I}^{K} X_{kj} = 0 \quad (19)$$

$$X_{kj} + \sum_{i=I}^{K} X_{ij} - X_{jk} - \sum_{i=I}^{K} X_{ji} = 0 \quad (20)$$

$$\text{for all } j = I, \ldots, K$$

$$\text{all } X_{ij} \geq 0 \quad (21)$$

Each such subproblem has (J + 2) equations and $((DJ)^2 + 2DJ)$ variables. Although we could solve each subproblem by a standard linear programming code, there is a special structure to the subproblem for-

mulation which can be exploited to reduce the solution time for each subproblem. For this purpose consider problem $P_k - 1$ below (where $C_{kk} = \infty$) which is the problem of finding the shortest path from depot k through the trans-shipment network and back to depot k. We can easily identify this shortest path by using the proposed shortest path algorithm [49]. Theorem 1 below shows how the solution to P_k can be obtained from the solution to $P_k - 1$, k = 1, ..., D. Note that equation (24) is really a combination of equations (18) and (19).

Problem $P_k - 1$:

$$\text{Min } f_k(X) = \sum_{j=I}^{K} \hat{C}^o_{kj} X_{kj} + \sum_{i=I}^{K} \sum_{j=I}^{K} \hat{C}_{ij} X_{ij} + \sum_{j=I}^{K} \hat{C}^r_{jk} X_{jk} \tag{22}$$

$$\text{subject to: } \sum_{j=I}^{K} X_{kj} = 1 \tag{23}$$

$$X_{kj} + \sum_{i=I}^{K} X_{ij} - X_{jk} - \sum_{i=I}^{K} X_{ji} = 0 \tag{24}$$

$$\text{for } j = k \text{ and } j = I, \ldots, K$$

$$\text{all } X_{ij} \geq 0 \tag{25}$$

The objective function coefficients in equation (22) may be such that there is a negative cycle in the network of problem $P_k - 1$. The Dantzig, Blattner, and Rao algorithm [49] can detect such a negative cycle, if one exists.

Theorem 1. Let $\hat{X}$ be a solution to problem $P_k - 1$ with $\hat{X}_{kp} = \hat{X}_{qk} = 1$, i.e., arcs (k,p) and (q,k) are in the shortest path. Then:

(i) If $f_k(\hat{X}) \geq 0$, then: $X^* = 0$ is the optimal solution to

problem P_k.

(ii) If $f_k(\hat{X}) < 0$ and finite, with $(k, p, i_1, i_2, \ldots, i_m, q, k)$ as the shortest path, then: the optimal solution to problem P_k is given by $X^*_{kp} = X^*_{pi_1} = \ldots = X^*_{i_m q} = X^*_{qk} = S_k$ and all other $X^*_{ij} = 0$.

(iii) If $f_k(\hat{X}) < 0$ and infinite, then let $i_{k_1}, i_{k_2}, \ldots, i_{k_m}, i_{k_1}$ be the cycle with negative length and $i_{k_j} \neq k$, $j = 1, \ldots, m$, then: $X^*_{k_1 k_2} = X^*_{k_2 k_3} = \ldots = X^*_{k_m k_1} = 1$ with all other $X^*_{ij} = 0$, is an extreme ray of problem P_k.

Proof.

(i) Clearly $X = 0$ is feasible to P_k. Suppose X^* be a solution to P_k with $f_k(X^*) < 0$. This implies there exists a negative cycle $i_1, i_2, \ldots, i_m, i_1$. Then $\hat{\hat{X}}_{i_1 i_2} = \ldots = \hat{\hat{X}}_{i_m i_1} = 1$ and all other $\hat{\hat{X}}_{ij} = 0$ is a feasible solution to $P_k - 1$ with $f_k(\hat{\hat{X}}) < 0$, this contradicts the assumption that $\hat{X}$ solves $P_k - 1$.

(ii) Suppose $X' \neq X^*$ is optimal to P_k, i.e., $f_k(X') < f_k(X^*)$. Let $X'_{ki_1} = X'_{i_1 i_2} = \ldots = X'_{i_m k} = S_k$. Then: $X''_{ki_1} = X''_{i_1 i_2} = \ldots = X''_{i_m k} = 1$ is a feasible to $P_k - 1$ with $f_k(X'') < f_k(\hat{X})$, a contradiction.

(iii) Substituting the solution $X^* \geq 0$ in equations (18), (19), and (20) we get:

$$\sum_{j=I}^{K} X^*_{kj} = 0$$

$$\sum_{j=I}^{K} X^*_{jk} - \sum_{j=I}^{K} X^*_{kj} = 0$$

$$X^*_{kj} + \sum_{i=I}^{K} X^*_{ij} - X^*_{jk} - \sum_{i=I}^{K} X^*_{ji} = 0 \quad \text{for all } j = I, \ldots, K$$

Hence, it is an extreme ray. This completes the proof.

It may be noted that case (iii) cannot arise under the assumptions of our problem, since t_j^o and t_j^r are fixed. Hence from the four conditions resulting in null variables discussed earlier, either $C_{jk} = \infty$ or $C_{kj} = \infty$ for each j, k = I, ..., K. On the other hand, suppose t_j^o is stated as an interval of time during which a job can start, i.e., the earliest starting time (t_j^E), the latest starting time (t_j^L), along with job duration time (d_j) is specified. In this situation case (iii) can occur.

It may be useful at this point to illustrate and discuss the three cases in Theorem 1. An example problem discussed later will contain each of the above cases. Consider iteration 1 of the example problem, the solution of P_2- 1 is the shortest path shown in figure 1.

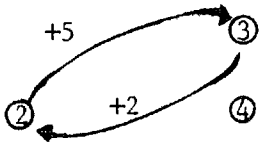

Figure 1. Illustration of case (i) of Theorem 1.

We can send one vehicle along this shortest path at a positive cost of 7. Hence, the solution to problem P_2 is all $X_{ij} = 0$, an example of case (i).

In iteration 2 we find that P_2-1 has identified a negative cycle with $f_2(X) = -\infty$ as shown in case (iii). The cycle is (3 - 4 - 3) which is diagrammed in Figure 2.

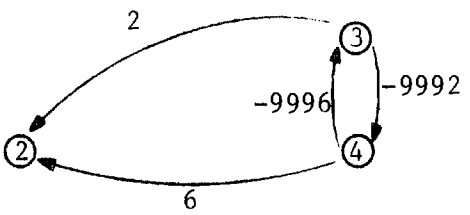

Figure 2. Illustration of case (iii) of Theorem 1.

Here the optimal solution to P_2 is unbounded. The extreme ray will have $X_{34} = X_{43} = 1$ and all other $X_{ij} = 0$.

Finally, case (ii) appears in iteration 3, where P_2-1 determines a shortest path with finite negative length, as shown in Figure 3 below.

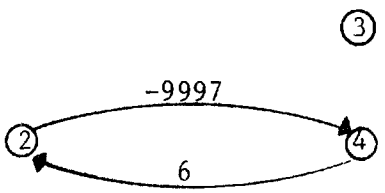

Figure 3. Illustration of case (ii) of Theorem 1.

Here we find the solution to P_2 is $X_{24} = X_{42} = 6$ and all other $X_{ij} = 0$.

From the above discussion, we see that in any finite solution to P_k, the value of each variable is either 0 or S_k. Likewise, in the case of an unbounded solution, the extreme ray will have either 0 or 1 as components. In the Dantzig-Wolfe decomposition procedure,

each such solution is converted into a column of the restricted master problem, if the solution has not already been considered and if it will improve the objective function. From equation (11) it will be seen that such a column will have an entry, a_j, in row j, $j = 1, \ldots, J$ as follows:

$$a_j = \begin{cases} 0 & \text{if case (i) of Theorem 1 arises.} \\ S_k & \text{if case (ii) of Theorem 1 arises and job j is in the shortest path for depot k.} \\ 1 & \text{if case (iii) of Theorem 1 arises and job j is in the cycle.} \end{cases}$$

Solution of the restricted master problem will yield a vector π, of dual variable values, which is used to modify the cost coefficients of the subproblems. Appendix B discusses how the π values can be calculated from the current tableau. The procedure terminates when no vector can be added to the restricted master problem which improves the objective function. At this point, the solution is in terms of λ_i^k, the convex combination of extreme points and positive linear combination of extreme rays. This is readily converted to the solution in terms of X_{ij}^k.

Now using the matrix notation from the beginning of this chapter we can summarize and outline the step by step procedure of our algorithm.

(i) Solve each subproblem $B_k X_k = b_k$ and obtain the shortest path or identify a negative cycle

(ii) Convert the subproblem solutions to columns of the restricted

master problem $\sum_{k=1}^{D} A_k X_i^k \lambda_i^k = b_o$; $\sum_{i=1}^{m} \lambda_i^k = 1$ for each $k = 1, \ldots, D$ where the coefficient of λ_i^k for the ith vector added is 1 for an extreme point solution X_i^k.

(iii) Solve the restricted master problem and obtain a vector of dual variable values, π.

(iv) Modify C_k by subtracting πA_k and resolve the subproblems.

(v) Stop when no new subproblem solutions can be added to the restricted master problem.

(vi) Apply heuristic programming to restore integrality if necessary.

Integer Solution

If a non-integer value appears in the optimal solution, the same value will appear on all arcs of a cycle from some depot k, through a series of one or more jobs and back to k. This is true since, in our stated problem, case (iii) of Theorem 1 cannot arise and the optimal solution is simply a convex combination of extreme points, each representing one cycle from a given depot and back.

The integrality along a cycle can be enforced either by increasing or decreasing the flow. If we reduce the flow, we will fail to meet the demands of the jobs along the cycle but we require that all demands be met in our problem statement. Initially we assumed that we had sufficient availability to meet all demands. Hence, we can increase the flow on all non-integer cycles to the next greater integer. This solution obviously can fail to be optimal since we have increased the costs and will be allowing excess vehicles at some jobs.

Further Analysis of Solution

Now if t_j^o is specified in terms of a time interval (along with a job duration time), case (iii) of Theroem 1 can occur. In this case we know t_j^E and t_j^L, the earliest and latest times when the job can start and also the job duration d_j. And the solution of problem P need not be optimal. To see this, note that constraint (11) of problem P permits a demand, M_j, to be met only from vehicles arriving from a depot or from another job. In other words, it does not permit a vehicle to meet a unit of demand at j and continue on to meet one or more units of demand at the same job site even though it can be done without violating the dynamic constraints. If this happens, the following heuristic procedure can be applied to reduce the value of the objective function.

(i) Let $i_1, i_2, \ldots, i_k, i_1$ denote a cycle in the solution with $X_{i_1 i_2} = X_{i_2 i_3} = \ldots = X_{i_k i_1} \geq K$. Let $X_{j i_1}^k > 0$ for $j \in S$ $(j \neq i_2, \ldots, i_k)$ for some k. Let d_i represent the job duration for meeting one unit of requirement at job i.

(ii) For each $X_{j i_1}^k$, $j \in S$, let t^r be its actual release time obtained from the schedule given by the algorithm. Let

$$z_j^{(1)} = \left[\frac{t_{i_2}^L - (t^r + t_{i_1 i_2} + h_{i_1 i_2})}{d_{i_1}} \right]$$

where $[\alpha]$ denotes the largest integer less than or equal to α.

(iii) Let $S_1 = \{j : j \in S \text{ and } z_j^{(1)} \geq 1\}$. Stop if $S_1 = \emptyset$ since this means we cannot reduce cycles without violating time

constraints. Otherwise, select a $q \in S_1$ and let $X^k_{qi_1} = p$. Now let $z^{(2)}_q = \text{Max } \{[K/p], 1\}$ where $[K/p]$ is the largest integer $\leq K/p$. If $K > p$, $z^{(2)}_q$ measures the maximum number of cycles that the p vehicles can perform. If $K \leq p$, $z^{(2)}_q = 1$ means K of the p vehicles can remove the cycle.

(iv) Let $z = \text{Min } (z^{(1)}_q, z^{(2)}_q)$. Decrease $X_{i_1 i_2}, \ldots, X_{i_k i_1}$ and K by $(z \cdot p)$ if $K > p$ and by $(z \cdot K)$ if $K \leq p$.

An Example

The following example is presented for the purpose of clarifying the solution procedure and is not to be considered as a realistic problem. This example will demonstrate the use of the Dantzig-Wolfe decomposition algorithm and Dantzig, Blattner, and Rao's algorithm for the shortest path problem. It is expected that the reader is familiar with these procedures, and therefore their details will be shown but not extensively treated. The example is designed so that case (iii) of Theorem 1 does arise. For obtaining this case, we had to express t^o_j as an interval, such that any starting time for job j was acceptable.

Consider a situation with two depots (#1 and #2) and two job locations (#3 and #4). The cost matrix for this case is:

	1	2	3	4
1	∞	∞	12	15
2	∞	∞	5	2
3	17	2	∞	7
4	8	6	3	∞

There are 15 and 6 vehicles available at depots 1 and 2 respectively. Job 3 requires 9 vehicles and job 4 requires 8. The situation is such that there are no compatibility restrictions and we are allowed to accomplish jobs 3 and 4 in either order without violating any time restrictions. This leads to the following mathematical statement of the problem according to our model in equations (7) through (12).

$$\text{Min } f(X) = 12X_{13}^1 + 15X_{14}^1 + 5X_{23}^2 + 2X_{24}^2 + 7X_{34}^1 + 7X_{34}^2 + 3X_{43}^1 + 3X_{43}^2 + 17X_{31}^1 + 8X_{41}^1 + 2X_{32}^2 + 6X_{42}^2$$

subject to:

$$\begin{aligned} X_{13}^1 + X_{14}^1 &\leq 15 \\ X_{31}^1 + X_{41}^1 - X_{13}^1 - X_{14}^1 &= 0 \\ X_{13}^1 + X_{43}^1 - X_{31}^1 - X_{34}^1 &= 0 \\ X_{14}^1 + X_{34}^1 - X_{41}^1 - X_{43}^1 &= 0 \end{aligned} \tag{26}$$

$$\begin{aligned} X_{23}^2 + X_{24}^2 &\leq 6 \\ X_{32}^2 + X_{42}^2 - X_{23}^2 - X_{24}^2 &= 0 \\ X_{23}^2 + X_{43}^2 - X_{32}^2 - X_{34}^2 &= 0 \\ X_{24}^2 + X_{34}^2 - X_{42}^2 - X_{43}^2 &= 0 \end{aligned} \tag{27}$$

$$\begin{aligned} X_{13}^1 + X_{23}^2 + X_{43}^1 + X_{43}^2 &= 9 \\ X_{14}^1 + X_{24}^2 + X_{34}^1 + X_{34}^2 &= 8 \end{aligned} \tag{28}$$

$$\text{all } X_{ij}^{k} \geq 0$$

The equations denoted by (26) and (27) correspond to subproblems one and two, respectively. Our first step is then to solve the subproblems by finding the shortest path through the cost matrix given above. Using Dantzig, Blattner, and Rao's algorithm (which can be visually verified for this trivial example) we get the shortest paths of 1 to 4 to 1 and 2 to 3 to 2. Since there is a positive cost associated with both of these routes we will send 0 vehicles along each route. This will add a zero vector to the restricted master problem corresponding to constraints (28) above. Since this is an extreme point solution there will be a 1 in the appropriate linear combination constraint equations. Here the coefficients in the objective function for the artificial variables were taken as 9999. Omitting the artificial variables for simplicity the restricted master problem will appear as follows.

Iteration 1:

ROW	VARIABLES		RHS
	α_1	β_1	
0	0	0	0
1	0	0	9
2	0	0	8
3	1	0	1
4	0	1	1

The resulting dual variable values are 9999 and 9999 corre-

sponding to the initial basic artificial variables of row 1 and 2, which relate to jobs 3 and 4, respectively. Now using these dual variable values we can modify the original cost matrix to obtain:

	1	2	3	4
1	∞	∞	-9987	-9984
2	∞	∞	-9994	-9997
3	17	2	∞	-9992
4	8	6	-9996	∞

Notice that a negative cycle has now been created between jobs 3 and 4. This will be detected by the shortest path algorithm and arises in both subproblems. Thus our solutions are 1 to 4 to 3 to 4 and 2 to 3 to 4 to 3. These negative cycles imply unbounded subproblem solutions and the cycles are extreme rays of the solution set. In this case, following the Dantzig-Wolfe decomposition procedure we must convert the extreme rays to columns of the restricted master problem without including the column as part of the linear combination constraints. The columns generated are:

10		10
1		1
1	and	1
0		0
0		0

Adding the equivalent of these we get the following initial simplex

tableau.

Iteration 2:

ROW	VARIABLES			RHS
	α_1	β_1	γ	
0	0	0	10	0
1	0	0	1	9
2	0	0	1	8
3	1	0	0	1
4	0	1	0	1

This iteration yields dual variable values of -9989 and 9999, which in turn give an adjusted cost matrix of:

	1	2	3	4
1	∞	∞	10001	-9984
2	∞	∞	9994	-9997
3	17	2	∞	-9992
4	8	6	9992	∞

The resulting shortest path solutions are 1 to 4 to 1 and 2 to 4 to 2 with finite negative objective function values. These routes convert to restricted master problem columns of:

345	48
0	0
15	6
1	0
0	1

Neither of these have been previously considered so we will add both to the restricted master problem and get:

Iteration 3:

ROW	VARIABLES					RHS
	α_1	β_1	γ	α_2	β_2	
0	0	0	10	345	48	0
1	0	0	1	0	0	9
2	0	0	1	15	6	8
3	1	0	0	1	0	1
4	0	1	0	0	1	1

Here the dual variable values are 9999 and -9989 which give the adjusted cost matrix:

	1	2	3	4
1	∞	∞	-9987	10004
2	∞	∞	-9994	9991
3	17	2	∞	9996
4	8	6	-9996	∞

This time we get shortest paths of 1 to 3 to 1 and 2 to 3 to 2 which generate the following columns:

435	42
15	6
0	0
1	0
0	1

Both columns are admissible and we get the following restricted master tableau.

Iteration 4:

ROW	VARIABLES							RHS
	α_1	β_1	γ	α_2	β_2	α_3	β_3	
0	0	0	10	345	48	435	42	0
1	0	0	1	0	0	15	6	9
2	0	0	1	15	6	0	0	8
3	1	0	0	1	0	1	0	1
4	0	1	0	0	1	0	1	1

This time we obtain dual variable values of 7 and 3 and the adjusted cost matrix is:

	1	2	3	4
1	∞	∞	5	12
2	∞	∞	-2	-1
3	17	2	∞	4
4	8	6	-4	∞

Here we get shortest paths of 1 to 3 to 4 to 1 and 2 to 4 to 3 to 2. The solution to subproblem 1 has a positive cost associated with it so this would generate a zero vector which we have already considered. Thus we need only add the column

42
6
6
0
1

Adding this column we get:

Iteration 5:

ROW	VARIABLES								RHS
	α_1	β_1	γ	α_2	β_2	α_3	β_3	β_4	
0	0	0	10	345	48	435	42	42	0
1	0	0	1	0	0	15	6	6	9
2	0	0	1	15	6	0	0	6	8
3	1	0	0	1	0	1	0	0	1
4	0	1	0	0	1	0	1	1	1

This results in dual variable values of 10 and 0 and an adjusted cost matrix of:

	1	2	3	4
1	∞	∞	2	15
2	∞	∞	-5	2
3	17	2	∞	7
4	8	6	-7	∞

and the shortest path of 1 to 3 to 4 to 1 has a positive cost and 2 to 3 to 2 has already been considered.

The pertinent information from the last restricted master problem is

$$\beta_3 = 1/6 \qquad \beta_4 = 5/6 \qquad \gamma = 3$$

This tells us to use $(1/6 \cdot 6) = 1$ vehicle along route 2 to 3 to 2 and similarly 5 vehicles along route 2 to 4 to 3 to 2 and 3 vehicles along cycle 3 to 4 to 3.

Now, we shall examine whether the solution can be improved as discussed earlier. The reduction of cycling is dependent on the operational constraints so let us consider two cases:

Case 1. Suppose $t_{34} = t_{43} = 20$ min. and $h_{34} = h_{43} = 5$ min. and $w_3 = w_4 = 10$ min. and $t_3^E = t_4^E = 8$ am with $t_3^L = t_4^L = 6$ pm with job durations at each job of one hour per load. Following our heuristic algorithm for breaking cycles, given earlier, we see that:

$$z_2^{(1)} = \left[\frac{6 \text{ pm} - (9 \text{ am} + 20 \text{ min} + 5 \text{ min})}{1 \text{ hour}}\right] = 8$$

and

$$z_2^{(2)} = \text{Max } \{ [3/5], 1 \} = 1$$

Hence we can break the cycle by retaining 3 vehicles at job 4 for one additional job duration.

Case 2. Suppose we alter the job duration of job 4 to 2 hours and t_3^L to 11 am and leave the other parameters as stated in Case 1. Here we calculate:

$$z_2^{(1)} = \left[\frac{11 \text{ am} - (10 \text{ am} + 20 \text{ min} + 5 \text{ min})}{2 \text{ hours}}\right] = 0$$

Hence, $S_1 = \emptyset$ and the cycle cannot be broken.

This completes the solution of our small example problem. The computer solutions of other more complex and larger problems are summarized in Table 4.

CHAPTER III

COMPUTATIONAL RESULTS AND CONCLUSIONS

The algorithm was programmed in FORTRAN for solution on the Univac 1108 computer. No special attempt was made at optimizing the program or to reduce storage requirements. As will be seen, the computation time is very encouraging, but the program takes a large amount of storage.

The general flow diagram of the algorithm code is shown in Figure 4. A main program was used to control the basic data input and the sequencing of the solution procedure. Subroutines were used for modifying the cost matrix in response to the operational constraints, for adding columns to the restricted master problem, for solving the restricted master problem by linear programming, and for solving the subproblems by a shortest path method.

First the main program reads in the number of jobs and depots and initializes some defining parameters. Then the unaltered cost matrix is read from data cards. If any modifications are required to satisfy operational constraints then subroutine MODIFY is called to accomplish them. Variable initialization is also accomplished at this stage. Next, we set up the penalized cost matrix for the subproblems and solve each subproblem by calling subroutine SPATH. The shortest path solutions are saved and a vector generated for possible inclusion in the restricted master problem. The initial tableau of

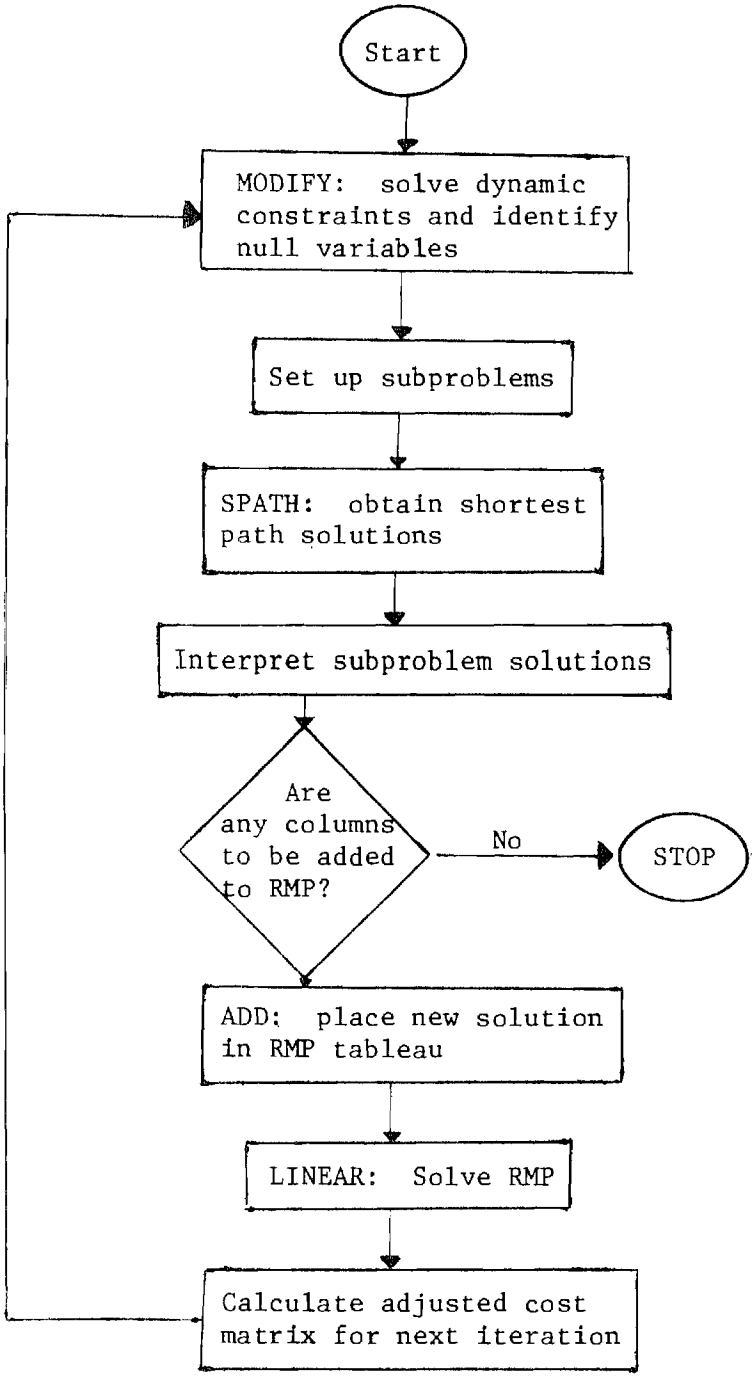

Figure 4. Flow Diagram of the Algorithm (for continuous solutions)

the restricted master problem is saved at each iteration and new columns added if appropriate by subroutine ADD. Each column of the restricted master tableau is matched with the saved shortest path which generated the column for ease in calculating the final solution. The restricted master problem is then solved by calling subroutine LINEAR. This provides a new set of dual values which are used to adjust the cost matrix for the subproblems. When no new columns can be added to the restricted master problem, the algorithm terminates and the main program calculates and prints the optimal solution.

Subroutine ADD takes a previously selected subproblem solution vector and converts it to a column of the restricted master problem to include the value of the linear combination component.

Subroutine MODIFY, if called, reads in the travel, waiting, and safety times and the compatibility conditions. The subroutine then checks to see if any of the operational constraints are violated and if so, causes the appropriate modification of the cost matrix to prevent the undesired combination.

The core of subroutine LINEAR is from the LINEAR-B program of Dr. Ronald L. Rardin (Georgia Institute of Technology). The subroutine uses a modified simplex method to solve a linear problem. The original program has been adjusted to perform minimization and to be compatible with the formats of this particular problem. New dual variable values are determined within the subroutine at each iteration and are transferred by a COMMON statement.

Subroutine SPATH finds the shortest path through the input

modified cost matrix. Using the methods and equations of the selected shortest path algorithm [49], negative cycles can be detected. A negative cycle implies an unbounded solution to the subproblem. The output of the subroutine is the shortest path (including any negative cycle found) and the total cost for one vehicle traveling that path.

All of the subroutines and the main program use basic FORTRAN techniques and are easily followed step by step. No special computer science techniques were utilized; nor are any required to work with this program.

The maximum size problem that the Univac 1108 can handle by our program is a 3-depot 18-job problem. This is because the program required an excess of 46 K of memory for data storage and manipulation. Through packing of matrices and the use of auxiliary tapes, the sizes of the problems handled can be substantially increased, and 10-depot 100-job problems will be within the capacity of the current approach. It may be noted that Gavish and Schweitzer [14] were able to achieve a significant reduction in computer time by reprogramming from FORTRAN H to ASSEMBLAR. It is anticipated that the application of similar techniques to this algorithm would achieve comparable results.

Test Problems

The following test problems in Table 4 were selected as typical of the model derived in this thesis. The number of jobs was varied in steps of three from 3 to 18. We elected to use only three

depots for the test problems since this provided a depot-to-job ratio varying from 1:1 to 1:6.

The cost data for each problem was generated by use of a multiplicative congruential random number generator, with the same random number seed used for each problem set. More details are given in Appendix A. Certain assumptions were made in the generation subroutine to make the formulation realistic.

First, we assumed that each depot had an equal density of jobs about it and that the vehicle availability at each depot exceeded the total demand of the jobs within that depot's area by 20 percent. Secondly, we assumed that the depot and job numbers provided a general measure of distance which could be used to realistically adjust the costs for the spread of the depots and jobs. Finally, we provided for an increase of return-to-depot costs ten percent of the time to account for overtime or other similar costs.

As stated in the example problem of Chapter II, the operational constraint aspects of this problem did not significantly affect the solution procedure. For this reason we have omitted the implementation of these constraints in our test problems. As discussed earlier, our problem has either $C_{ij} = \infty$ or $C_{ji} = \infty$, which was insured in our data generation method.

Table 4 lists the details of the problems solved and computational results. The column R. N. Seed gives the random number seed for the purpose of reproducing these problems if necessary.

Table 4. Test Problem Solutions

Problem	No. of Depots	No. of Jobs	R. N. Seed	Total No. Cols. in RMP	No. Basic Cols. in Opt. Soln.	Max. Job/Route	Execution Time (Sec.)
1	3	3	9573	11	3	1	.19
2	3	6	9573	24	6	2	.77
3	3	9	9573	24	9	3	1.55
4	3	12	9573	76	11	4	9.75
5	3	15	9573	79	13	3	14.10
6	3	18	9573	89	10	4	24.34
7	3	3	8537	14	3	1	.31
8	3	6	8537	29	4	3	1.05
9	3	9	8537	23	7	2	1.14
10	3	12	8537	106	9	4	27.05
11	3	15	8537	46	12	3	6.74
12	3	18	8537	101	15	3	32.16
13	3	3	9217	12	4	1	.25
14	3	6	9217	33	6	3	1.63
15	3	9	9217	34	9	5	2.28
16	3	12	9217	25	9	2	2.10
17	3	15	9217	40	12	3	5.80
18	3	18	9217	84	12	3	27.13
19	3	3	8991	12	3	1	.30
20	3	6	8991	87	6	2	6.69
21	3	9	8991	39	7	3	2.44
22	3	12	8991	37	8	3	4.07
23	3	15	8991	72	12	4	12.92
24	3	18	8991	38	15	2	5.93

Conclusions

Most of the published studies do not consider the requirement that a vehicle return to its depot of origin. By considering a rather general model with this added constraint, this study hopefully represents an advancement in this area of research. Also, through the use of the Dantzig-Wolfe decomposition procedure and a shortest path algorithm it is shown that reasonably sized problems can be solved effectively.

In applying the proposed solution technique, computer memory limitations were encountered. This forced us to work with relatively small problems, but the results of Table 4 show that the computation times are relatively small. As discussed earlier, more advanced programming methods should be able to solve larger problems.

It may be noted from Table 4 that problem 10 took substantially more time to solve than problems 4, 16, and 22 for the same number of depots and jobs. The total number of columns considered in the restricted master problem was also significantly higher in this case. The increased solution time probably came from the repeated execution of LINEAR or SPATH subroutines.

We discussed in some detail the implications of extending our model to incorporate the specification of a starting interval and job duration for job j, in lieu of the fixed starting time t_j^o. Other manipulations of the dynamic constraint parameters can be handled with less difficulty, e.g., any particular routing may be prohibited simply by setting the appropriate compatibility indices to zero.

Recommendations

This research has pointed to areas where further work is indicated. Effort expended on the following extensions would improve the applicability of this algorithm to the transportation field.

(i) How can we minimize the size of the fleet without increasing the overall costs?

(ii) Can the fleet be redistributed among the depots to further reduce the operating costs?

(iii) How can compatibility for a series of three or more jobs be expressed?

(iv) How can re-dispatch of a vehicle be allowed simultaneously with consideration of arrival intervals?

(v) What programming techniques can be applied to this FORTRAN program in order to reduce the core storage requirement?

(vi) Can the dynamic aspects of the problem be included in the analytical model and an optimal integer solution achieved?

These areas in no way imply inadequacy of the present research, but are simply beyond the scope of this work.

APPENDIX A

Main Program User's Instructions

The program is written to accept punch cards in the following order and format:

(i) Number of depots, number of jobs (integers)

(ii) One card for each depot containing the depot to job costs in sequence separated by commas, follwed by the availability at that depot (floating point)

(iii) One card for each job containing the job to depot and then the job to job costs all in sequence, followed by the demand for that job (floating point)

(iv) A one integer code, when set to zero, will prevent compatibility modifications. If set to an integer other than zero, MODIFY will be called to make the desired changes. (integer)

If subroutine MODIFY is to be used, additional data cards are required as follows:

(v) Number of incompatible entries (integer)

(vi) One card for each incompatible entry giving the pair of ordered job numbers which are not compatible separated by commas. (integers)

(vii) One card for each depot giving the travel time to each

job in sequence separated by commas (floating point)

(viii) One card for each job giving the travel time to each depot and then to each job in sequence separated by commas followed by the start and release times for that job (floating point)

(ix) One card for each depot giving the safety time to each job in sequence separated by commas (floating point)

(x) One card for each job giving the safety time to each depot and then to each job in sequence separated by commas followed by the maximum waiting time allowed at that job (floating point)

To clarify the use of the data cards the following set would be required to run the example problem of Chapter II.

```
2, 2
12., 15., 15.
5., 2., 6.
17., 2., 9999., 7., 9.
8., 6., 3., 9999., 8.
0
```

The output of the program states how many vehicles to send on a particular route. Thus the dispatcher has only to assign a certain number of his vehicles to the specified route. A sample output line would read, "Send 8. vehicles along route 1, 4, 3, 1," which means

that 8 vehicles are to be sent from depot 1 to job 4, then on to job 3 and finally back to depot 1.

MAIN PROGRAM FORTRAN

SOURCE CODE LISTING

```
FELCH-JHON-E*TRANS.MAIN
      1       C
      2       C*****DATA READ-IN AND INITIALIZATION                                  ******
      3       C
      4             COMMON C(150),CSAVE(150),COST(25,26),DUAL(45),LABEL(25,150),
      5            1NVECT(5),RMP(45,150),VECTOR(50,150),D,I,J,K,L,M,JM,JN,NTOT,ISHORT(
      6            222),COSTM(25,26),NMIT,CTOTAL,CYNEG
      7             DIMENSION B(45,150)
      8             DIMENSION ISV(5,30,22)
      9             INTEGER LABEL,D
     10             READ(5,10)D,J
     11          10 FORMAT( )
     12             I=D+1
     13             K=D+J
     14             L=K+1
     15             M=K+2
     16             JI=2*J+2
     17             JK=2*J+D
     18             JL=JK+1
     19             JM=J+1
     20             JN=2*J
     21             JO=2*J+1
     22          20 DO 30 II=1,D
     23             NVECT(II)=0
     24          30 CONTINUE
     25             NMIT=I
     26             DO 40 IJ=1,J
     27             DUAL(IJ)=0.
     28          40 CONTINUE
     29             DO 60 II=1,D
     30             DO 50 IJ=1,D
     31             COST(II,IJ)=9999.
     32          50 CONTINUE
     33          60 CONTINUE
     34             DO 70 II=1,D
     35             READ(5,10)(COST(II,IJ),IJ=I,L)
     36          70 CONTINUE
     37             DO 80 II=I,K
     38             READ(5,10)(COST(II,IJ),IJ=1,L)
     39          80 CONTINUE
     40             READ(5,10) NN
     41             IF(NN.EQ.0) GO TO 85
     42             CALL MODIFY
     43          85 DO 90 II=I,K
     44             IK=II-D
     45             DO 89 IJ=I,K
     46             COSTM(II,IJ)=COST(II,IJ)-DUAL(IK)
     47             IF(II.EQ.IJ) COSTM(II,IJ)=9999.
     48          89 CONTINUE
     49          90 CONTINUE
     50       C
     51       C*****SOLUTION OF SUBPROBLEMS                                          ******
     52       C
     53         100 DO 180 II=1,D
     54             DO 110 IJ=I,K
     55             IK=IJ-D
     56             COSTM(L,IJ)=COST(II,IJ)-DUAL(IK)
```

```
57            COSTM(IJ,M)=COST(IJ,II)
58            COSTM(M,IJ)=9999.
59            COSTM(IJ,L)=9999.
60          110 CONTINUE
61            COSTM(L,L)=9999.
62            COSTM(L,M)=9999.
63            COSTM(M,L)=9999.
64            COSTM(M,M)=9999.
65            DO 130 IJ=1,JM
66            VECTOR(II,IJ)=0.
67          130 CONTINUE
68          140 CALL SPATH(II)
69            IK=J+2
70            DO 145 IJ=1,IK
71            ISV(II,NMIT,IJ)=ISHORT(IJ)
72          145 CONTINUE
73            DO 160 IJ=I,K
74            IK=IJ-J
75            IF(ISHORT(2).EQ.IJ.AND.CYNEG.GE.0.) VECTOR(II,IK)=COST(II,L)
76            IR=J+2
77            DO 150 IL=3,IR
78            IO=ISHORT(IL)
79            IF(IO.EQ.II.OR.IO.EQ.0) GO TO 160
80            IF(IO.EQ.IJ) VECTOR(II,IK)=VECTOR(II,IK)+COST(I,L)
81          150 CONTINUE
82          160 CONTINUE
83            VECTOR(II,JM)=CTOTAL*COST(II,L)
84          180 CONTINUE
85        C
86        C*****SET UP OF INITIAL RESTRICTED MASTER PROBLEM                    ******
87        C
88            IF(NMIT.GT.I) GO TO 300
89            DO 220 II=1,150
90            DO 200 IJ=1,I
91            LABEL(IJ,II)=0
92            RMP(IJ,II)=0.
93          200 CONTINUE
94            DO 210 IJ=I,K
95            RMP(IJ,II)=0.
96          210 CONTINUE
97          220 CONTINUE
98            DO 230 II=1,150
99            CSAVE(II)=0.
100         230 CONTINUE
101           NTOT=1
102           DO 240 II=1,D
103           CALL ADD(II)
104           NTOT=NTOT+1
105         240 CONTINUE
106       C
107       C*****ADDITION OF NEW SUBPROBLEM VECTORS TO THE RESTRICTED           ******
108       C*****MASTER PROBLEM                                                ******
109       C
110         300 CONTINUE
111           NTOT=1
112           DO 310 II=1,D
113           NTOT=NTOT+NVECT(II)
```

```
      310 CONTINUE
          NADD=0
          DO 350 II=1,D
          KTEST=VECTOR(II,JM)*100000
          IF(KTEST.EQ.0) GO TO 350
          NT=NTOT-1
          DO 340 IJ=1,NT
          IF(LABEL(II,IJ).EQ.0) GO TO 340
          IK=J+II
          IF(RMP(IK,IJ).LE.0..AND.CYNEG.GE.0.) GO TO 340
          IF(RMP(IK,IJ).GE.1..AND.CYNEG.LT.0.) GO TO 340
          DO 320 IK=1,J
          ITST=RMP(IK,IJ)*100000
          JTST=VECTOR(II,IK)*100000
          IF(ITST.NE.JTST) GO TO 340
      320 CONTINUE
          IF(VECTOR(II,JM).GE.CSAVE(IJ)) GO TO 350
      340 CONTINUE
          CALL ADD(II)
          NTOT=NTOT+1
          NADD=NADD+1
      350 CONTINUE
          IF(NADD.EQ.0.AND.NMIT.GT.I) GO TO 2000
C
C*****ADDITION OF ARTIFICIAL VARIABLES AND SOLUTION OF THE          ******
C*****RESTRICTED MASTER PROBLEM                                     ******
C
          IP=NTOT+K-1
          DO 420 II=NTOT,IP
          CSAVE(II)=9999.
          IK=II-NTOT+1
          DO 400 IJ=1,K
          RMP(IJ,II)=0.
      400 CONTINUE
          RMP(IK,II)=+1.
          DO 410 IJ=1,I
          LABEL(IJ,II)=99
      410 CONTINUE
      420 CONTINUE
          IRHS=IP+1
          DO 430 II=1,J
          IJ=II+D
          RMP(II,IRHS)=COST(IJ,L)
      430 CONTINUE
          DO 450 II=JM,K
          RMP(II,IRHS)=1.
      450 CONTINUE
      500 DO 520 II=1,K
          DO 510 IJ=1,IRHS
          B(II,IJ)=RMP(II,IJ)
      510 CONTINUE
      520 CONTINUE
          DO 530 II=1,IP
          C(II)=CSAVE(II)
      530 CONTINUE
          CALL LINEAR(K,IP,B,NMIT,45,150)
          NMIT=NMIT+1
```

```
      C
      C*****MODIFICATION OF THE TRANS SHIPMENT COST MATRIX                  ******
      C
            DO 610 II=I,K
            DO 600 IK=I,K
            IJ=IK-D
            COSTM(II,IK)=COST(II,IK)-DUAL(IJ)
            IF(II.EQ.IK) COSTM(II,IK)=9999.
        600 CONTINUE
        610 CONTINUE
            GO TO 100
       2000 ITFIN=NMIT-1
            DO 2003 II=1,D
            DO 2001 IJ=1,NTOT
            IF(LABEL(II,IJ).NE.1) GO TO 2001
            X=COST(II,L)*VECTOR(ITFIN,IJ)
            IY=LABEL(I,IJ)
            IZ=J+2
            WRITE(6,2002) X,(ISV(II,IY,IK),IK=1,IZ)
       2002 FORMAT(14X,'SEND ',1F5.2,' VEHICLES ALONG ROUTE ',8I3,1(/),19X,17I
           13,1(/))
       2001 CONTINUE
       2003 CONTINUE
            WRITE(6,2004)
       2004 FORMAT('1')
            END
```

```
FELCH-JHON-E*TRANS.ADD
     1            SUBROUTINE ADD(MJ)
     2            COMMON C(150),CSAVE(150),COST(25,26),DUAL(45),LABEL(25,150),
     3           1NVECT(5),RMP(45,150),VECTOR(50,150),D,I,J,K,L,M,JM,JN,NTOT,ISHORT(
     4           222),COSTM(25,26),NMIT,CTOTAL,CYNEG
     5            INTEGER LABEL,D
     6            DO 12 MM=1,D
     7            LABEL(MM,NTOT)=0
     8         12 CONTINUE
     9            CSAVE(NTOT)=VECTOR(MJ,JM)
    10            LABEL(MJ,NTOT)=1
    11            LABEL(I,NTOT)=NMIT
    12            DO 42 MM=1,J
    13            RMP(MM,NTOT)=VECTOR(MJ,MM)
    14         42 CONTINUE
    15            MN=J+MJ
    16            RMP(MN,NTOT)=1.
    17            DO 52 MM=1,D
    18            MP=J+MM
    19            IF(MP.EQ.MN.AND.CYNEG.GE.0.) GO TO 52
    20            RMP(MP,NTOT)=0.
    21         52 CONTINUE
    22            NVECT(MJ)=NVECT(MJ)+1
    23            RETURN
    24            END
```

```
FELCH-JHON-E*TRANS,MODIFY
      1              SUBROUTINE MODIFY
      2              COMMON C(150),CSAVE(150),COST(25,26),DUAL(45),LABEL(25,150),
      3             1NVECT(5),RMP(45,150),VECTOR(50,150),D,I,J,K,L,M,JM,JN,NTOT,ISHORT(
      4             222),COSTM(25,26),NMIT,CTOTAL,CYNEG
      5              DIMENSION HANDW(25,26),ICOMP(25,25),TIME(25,27)
      6              INTEGER LABEL,D
      7           10 FORMAT( )
      8              DO 30 II=1,K
      9              DO 20 IJ=1,K
     10              ICOMP(II,IJ)=1
     11           20 CONTINUE
     12           30 CONTINUE
     13              READ(5,10) N
     14              IF(N.EQ.0) GO TO 65
     15              DO 60 II=1,N
     16              READ(5,10) IK,IL
     17              ICOMP(IK,IL)=0
     18           60 CONTINUE
     19           65 DO 90 II=1,K
     20              DO 80 IJ=1,M
     21              TIME(II,IJ)=0.
     22           80 CONTINUE
     23           90 CONTINUE
     24              DO 100 II=1,D
     25              READ(5,10)(TIME(II,IJ),IJ=I,K)
     26          100 CONTINUE
     27              DO 110 II=I,K
     28              READ(5,10)(TIME(II,IJ),IJ=1,M)
     29          110 CONTINUE
     30              DO 170 II=1,K
     31              DO 160 IJ=1,L
     32              HANDW(II,IJ)=0.
     33          160 CONTINUE
     34          170 CONTINUE
     35              DO 180 II=1,D
     36              READ(5,10)(HANDW(II,IJ),IJ=I,K)
     37          180 CONTINUE
     38              DO 190 II=I,K
     39              READ(5,10)(HANDW(II,IJ),IJ=1,L)
     40          190 CONTINUE
     41              DO 200 II=1,K
     42              DO 210 IJ=1,K
     43              IF(ICOMP(II,IJ).EQ.0) COST(II,IJ)=9999.
     44              IF(TIME(II,M)+TIME(II,IJ)-TIME(IJ,L)+HANDW(IJ,L).LE.0) COST(II,IJ)
     45             1=9999.
     46              IF(TIME(II,M)+TIME(II,IJ)+HANDW(II,IJ)-TIME(IJ,L).GE.0) COST(II,IJ
     47             1)=9999.
     48          210 CONTINUE
     49          200 CONTINUE
     50              RETURN
     51              END
```

```
FELCH-JHON-E*TRANS.LINEAR
      1              SUBROUTINE LINEAR(NOROW,NOCOL,A,II,IDIM,JDIM)
      2              COMMON C(150),CSAVE(150),COST(25,26),DUAL(45),LABEL(25,150),
      3             1NVECT(5),RMP(45,150),VECTOR(50,150),D,I,J,K,L,M,JM,JN,NTOT,ISHORT(
      4             222),COSTM(25,26),NMIT,CTOTAL,CYNEG
      5              INTEGER LABEL,D
      6              NORHS=NOCOL+1
      7              DIMENSION A(IDIM,JDIM),IBV(45),KBV(45),COLIN(45)
      8              EPS=.000001
      9              MAXCOL=150
     10              MAXROW=45
     11              IF(NOROW.GT.MAXROW.OR.NOCOL.GT.MAXCOL) GO TO 910
     12              C(NORHS)=0.
     13              DO 150 NR=1,NOROW
     14              IF(A(NR,NORHS).LT.0.) GO TO 920
     15              IBV(NR)=0
     16        150 CONTINUE
     17       C-----GET INITIAL BASIS
     18              NC=NORHS
     19              DO 190 NI=1,NOROW
     20        170 NC=NC-1
     21              IF(NC.LE.0) GO TO 192
     22              NOZER=0
     23              NOONE=0
     24              DO 180 NR=1,NOROW
     25              IF(A(NR,NC).GT.-EPS.AND.A(NR,NC).LT.EPS) NOZER=NOZER+1
     26              IF(A(NR,NC).LE.1.-EPS.OR.A(NR,NC).GE.1.+EPS) GO TO 180
     27              NOONE=NOONE+1
     28              NOHIT=NR
     29        180 CONTINUE
     30              IF(NOONE.NE.1.OR.NOZER.NE.NOROW-1) GO TO 170
     31              IF(IBV(NOHIT).NE.0) GO TO 170
     32              KBV(NOHIT)=NC
     33              COLIN(NOHIT)=C(NC)
     34              IBV(NOHIT)=NC
     35        190 CONTINUE
     36        192 NOBSW=0
     37              DO 195 NR=1,NOROW
     38              IF(IBV(NR).NE.0) GO TO 195
     39              NOBSW=1
     40              WRITE(6,193) NR
     41        193 FORMAT('0 NO STARTING BASIC SOLN FOR ROW',I5)
     42        195 CONTINUE
     43              IF(NOBSW.EQ.1) GO TO 995
     44       C-----ELIMINATE BASIC VARIABLES FROM OBJECTIVE FUNCTION
     45              DO 197 NC=1,NORHS
     46              DO 196 NR=1,NOROW
     47        196 C(NC)=C(NC)-COLIN(NR)*A(NR,NC)
     48        197 C(NC)=+C(NC)
     49              NOITER=-1
     50       C-----BEGIN MAIN ITERATION LOOP--PRINT STATUS
     51        200 NOITER=NOITER+1
     52       C-----CHECK OPTIMALITY AND-OR FIND INCOMING COLUMN
     53        250 INCOL=0
     54              CMIN=-EPS
     55              DO 260 NC=1,NOCOL
     56              IF(C(NC).GE.CMIN) GO TO 260
```

```
57              CMIN=C(NC)
58              INCOL=NC
59            260 CONTINUE
60              IF(INCOL.EQ.0) GO TO 800
61          C-----PICK ROW TO PIVOT ON
62              INROW=0
63              RATMIN=999999.
64              DO 280 NR=1,NOROW
65              IF(A(NR,INCOL).LE.EPS) GO TO 280
66              RATIO=A(NR,NORHS)/A(NR,INCOL)
67              IF(RATIO.GE.RATMIN) GO TO 280
68              RATMIN=RATIO
69              INROW=NR
70            280 CONTINUE
71              IF(INROW.NE.0) GO TO 300
72          C-----UNBOUNDED SOLUTION
73              WRITE(6,285) INCOL
74            285 FORMAT('0 SOLUTION UNBOUNDED--ADDING COL',I5)
75              GO TO 995
76          C-----PIVOT
77            300 NC=KBV(INROW)
78              KBV(INROW)=INCOL
79              DO 305 NR=1,NOROW
80            305 COLIN(NR)=A(NR,INCOL)
81              CSTIN=C(INCOL)
82              COEF=A(INROW,INCOL)
83              DO 330 NC=1,NORHS
84              A(INROW,NC)=A(INROW,NC)/COEF
85              CORR=A(INROW,NC)
86              DO 310 NR=1,NOROW
87              IF(NR.EQ.INROW) GO TO 310
88              A(NR,NC)=A(NR,NC)-COLIN(NR)*CORR
89            310 CONTINUE
90              C(NC)=C(NC)-CSTIN*CORR
91            330 CONTINUE
92          C-----END MAIN ITERATION LOOP
93              GO TO 200
94            800 DO 840 IL=1,NOCOL
95              DO 820 IM=1,NOROW
96              IF(IL.EQ.KBV(IM)) GO TO 810
97              NBASIC=1
98              GO TO 820
99            810 VECTOR(II,IL)=A(IM,NORHS)
100             NBASIC=0
101             GO TO 830
102           820 CONTINUE
103           830 IF(NBASIC.EQ.0) GO TO 840
104             VECTOR(II,IL)=0.
105           840 CONTINUE
106             DO 850 IN=1,NOROW
107             ND=IBV(IN)
108             DUAL(IN)=9999.-C(ND)
109           850 CONTINUE
110           855 GO TO 999
111         C-----ERROR ROUTINES
112           910 WRITE(6,911)
113           911 FORMAT('0 TOO MANY ROWS OR COLUMNS')
```

```
      GO TO 995
  920 WRITE(6,921) NR
  921 FORMAT('0 NEGATIVE RHS IN ROW',I5)
      GO TO 995
  995 WRITE(6,996)
  996 FORMAT('0*****RUN ABORTED*****')
      WRITE(6,12) II
   12 FORMAT(T3,1I3)
      STOP
  999 RETURN
      END
```

```
FELCH-JHON-E*TRANS.SPATH
      1              SUBROUTINE SPATH(INDX)
      2              COMMON C(150),CSAVE(150),COST(25,26),DUAL(45),LABEL(25,150),
      3             1INVECT(5),RVP(45,150),VECTOR(50,150),D,I,J,K,L,M,JM,JN,NTOT,ISHORT(
      4             222),COSTM(25,26),NMIT,CTOTAL,CYNEG
      5              DIMENSION SLABEL(25),ILABEL(25),IPERM(25),IWAIT(27),DELTA(27,27)
      6              ISTART=L
      7              IEND=M
      8              CYNEG=0.
      9            4 IISAVE=9999
     10              SCHECK=9999.
     11              DO 14 II=I,M
     12              SLABEL(II)=COSTM(ISTART,II)
     13              ILABEL(II)=ISTART
     14              IPERM(II)=0
     15              IF(SLABEL(II).GE.SCHECK) GO TO 14
     16              SCHECK=SLABEL(II)
     17              IISAVE=II
     18           14 CONTINUE
     19              ILABEL(ISTART)=9999
     20              IPERM(ISTART)=1
     21              ICOUNT=1
     22           24 IPERM(IISAVE)=1
     23              ICOUNT=ICOUNT+1
     24              DO 44 II=I,M
     25              DO 34 IJ=I,M
     26              DELTA(II,IJ)=0.
     27           34 CONTINUE
     28           44 CONTINUE
     29              DO 54 II=I,M
     30              IWAIT(II)=0
     31           54 CONTINUE
     32              IQADD=IISAVE
     33              IWAIT(IISAVE)=1
     34              IWAIT(ISTART)=1
     35           64 DMIN=0.
     36              DO 74 II=I,M
     37              IF(IPERM(II).EQ.1) GO TO 74
     38              CHK=COSTM(IISAVE,II)+SLABEL(IISAVE)
     39              IF(CHK.GE.SLABEL(II)) GO TO 74
     40              SLABEL(II)=CHK
     41              ILABEL(II)=IISAVE
     42           74 CONTINUE
     43              DO 94 II=I,M
     44              DO 84 IJ=I,M
     45              IF(IWAIT(II).EQ.1.AND.IPERM(IJ).EQ.1.AND.IWAIT(IJ).EQ.0)
     46             1DELTA(II,IJ)=SLABEL(II)+COSTM(II,IJ)-SLABEL(IJ)
     47              DTEST=DMIN-.0005
     48              IF(DELTA(II,IJ).GE.DTEST) GO TO 84
     49              DMIN=DELTA(II,IJ)
     50              MINT=II
     51              MINL=IJ
     52           84 CONTINUE
     53           94 CONTINUE
     54              IF(DMIN.GE.0.) GO TO 98
     55              IWAIT(MINL)=1
     56              SLABEL(MINL)=SLABEL(MINT)+COSTM(MINT,MINL)
```

```
57              ILABEL(MINL)=MINT
58              CYNEG=SLABEL(MINL)+COSTM(MINL,IQADD)-SLABEL(IQADD)
59              IF(CYNEG.LT.0.) GO TO 96
60              IISAVE=MINL
61              DO 95 II=I,M
62              DELTA(II,MINL)=0.
63           95 CONTINUE
64              GO TO 64
65           96 SLABEL(IQADD)=COSTM(MINL,IQADD)+SLABEL(MINL)
66              ILABEL(IQADD)=MINL
67              ILOOP=IQADD
68              GO TO 105
69           98 II=J+2
70              IF(ICOUNT.EQ.II) GO TO 104
71              SCHECK=9999.
72              IISAVE=9999
73              DO 99 II=1,M
74              IF(IPERM(II).EQ.1) GO TO 99
75              IF(SLABEL(II).GE.SCHECK) GO TO 99
76              SCHECK=SLABEL(II)
77              IISAVE=II
78           99 CONTINUE
79              GO TO 24
80          104 ILOOP=IEND
81          105 IJ=J+2
82              DO 114 II=IJ,2,-1
83              ISHORT(II)=ILOOP
84              IILAST=II
85              ILOOP=ILABEL(ILOOP)
86              IF(CYNEG.LT.0..AND.ILOOP.EQ.IQADD) GO TO 115
87              IF(CYNEG.GE.0..AND.ILOOP.EQ.ISTART) GO TO 115
88          114 CONTINUE
89          115 ISHORT(1)=INDX
90              IK=IILAST
91              IF(CYNEG.LT.0.) IK=IILAST-1
92              IF(CYNEG.LT.0.) ISHORT(IK)=IQADD
93              DO 120 II=2,IJ
94              IILAST=II
95              ISHORT(II)=ISHORT(IK)
96              IF(IK.EQ.IJ) GO TO 122
97              IK=IK+1
98          120 CONTINUE
99          122 IF(CYNEG.GE.0.) ISHORT(IILAST)=INDX
100         124 IJ=ISHORT(2)
101             CTOTAL=COST(INDX,IJ)
102             DO 128 IJ=3,IILAST
103             IK=IJ-1
104             IY=ISHORT(IJ)
105             IZ=ISHORT(IK)
106             CTOTAL=CTOTAL+COST(IZ,IY)
107         128 CONTINUE
108             IF(CYNEG.GE.0.) GO TO 138
109         135 CTOTAL=0.
110             DO 136 IJ=2,IILAST
111             IU=IJ+1
112             IV=ISHORT(IJ)
113             IW=ISHORT(IU)
```

```
114             CTOTAL=CTOTAL+COST(IV,IW)
115             IF(IW.EQ.ILOOP) GO TO 138
116         136 CONTINUE
117         138 CONTINUE
118             II=IILAST+1
119             IK=J+2
120             DO 144 IJ=II,IK
121             ISHORT(IJ)=0
122         144 CONTINUE
123             IF(ISHORT(2).EQ.INDX) CTOTAL=0.
124             IF(CYNEG.LT.0.) RETURN
125             IF(CYNEG.GE.0..AND.SLABEL(IEND).LE.0.) RETURN
126             ISHORT(2)=INDX
127             ISHORT(3)=INDX
128             CTOTAL=0.
129             RETURN
130             END
```

Data Generation User's Instructions

The test data generation was accomplished by the following two subroutines being incorporated into the main program. The 70 and 80 DO-loops of MAIN were replaced by a CALL GENERA card.

The only data cards now required are:

(i) The number of depots, the number of jobs as integers separated by a comma.

(ii) The random number seed, which must be a four digit odd integer.

(iii) A code card with 0 in the first column to inactivate subroutine MODIFY.

DATA GENERATION SUBROUTINES

FORTRAN SOURCE CODE LISTING

```
FELCH-JHON-E*TRANS.GENERA
      1              SUBROUTINE GENERA
      2              COMMON C(150),CSAVE(150),COST(25,26),DUAL(45),LABEL(25,150),
      3             1NVECT(5),RMP(45,150),VECTOR(50,150),D,I,J,K,L,M,JM,JN,NTOT,ISHORT(
      4             222),COSTM(25,26),NMIT,CTOTAL,CYNEG
      5              INTEGER LABEL,D
      6              READ(5,11)ISEED
      7           11 FORMAT( )
      8              IC=J/D
      9              NTC=I
     10              DO 41 II=1,D
     11              INTAL=0
     12              DO 31 IJ=1,IC
     13              DO 21 IK=1,D
     14              DIST=(II-IK)*50
     15              IF(DIST.LT.0.) DIST=-DIST
     16              INT=50.*DRAND(ISEED)+DIST
     17              IF(INT.LT.0) INT=-INT
     18              COST(IK,NTC)=INT
     19              IF(DRAND(ISEED).LT..1) INT=INT+5
     20              COST(NTC,IK)=INT
     21           21 CONTINUE
     22              INT=10.*DRAND(ISEED)
     23              IF(INT.LT.0) INT=-INT
     24              COST(NTC,L)=INT
     25              INTAL=INTAL+INT
     26              NTC=NTC+1
     27           31 CONTINUE
     28              TAL=INTAL
     29              TAL=TAL+.2*TAL
     30              INTAL=TAL
     31              COST(II,L)=INTAL
     32           41 CONTINUE
     33              DO 61 II=I,K
     34              DO 51 IJ=I,K
     35              IF(II.GT.IJ) GO TO 51
     36              IK=II-IJ
     37              IF(IK.LT.0) IK=-IK
     38              IF(IK.LE.IC) IK=0
     39              DIST=IK*50
     40              INT=30.*DRAND(ISEED)+DIST
     41              IF(INT.LT.0) INT=-INT
     42              COST(II,IJ)=INT
     43              COST(IJ,II)=9999.
     44           51 CONTINUE
     45           61 CONTINUE
     46              WRITE(6,71) D,J
     47           71 FORMAT('1',9(/),14X,'A ',1I1,' DEPOT VEHICLE ASSIGNMENT PROBLEM WI
     48             1TH ',1I2,' JOB LOCATIONS',2(/),14X,'INITIAL COST MATRIX',1(/))
     49              WRITE(6,72)((II),II=1,K)
     50           72 FORMAT(14X,11I5,1(/))
     51              DO 74 II=1,K
     52              WRITE(6,73) II,(COST(II,IJ),IJ=1,K)
     53           73 FORMAT(14X,1I2,11F5.0,1(/),16X,11F5.0)
     54           74 CONTINUE
     55              WRITE(6,75)
     56           75 FORMAT('0',13X,'AVAILABILITIES AND DEMANDS')
```

```
57              WRITE(6,76)(COST(II,L),II=1,K)
58           76 FORMAT(16X,11F5.0)
59              RETURN
60              END
```

```
FELCH-JHON-E*TRANS.DRAND
      1              FUNCTION DRAND(ISEED)
      2              ISEED=ISEED*131075
      3              IF(ISEED.LE.0) ISEED=ISEED+34359738741
      4              DRAND=ISEED*.2910383E-10
      5              RETURN
      6              END
```

APPENDIX B

Dual Value Calculations

This appendix will examine the means of calculating the restricted master dual variable vector, π. Consider a basis, B, and let $A = [B|N]$. Partitioning C and X vectors, we get the problem in the following form:

$$\text{Min} \quad Z_o = C_B X_B + C_N X_N$$

$$\text{s.t.} \quad BX_B + NX_N = b$$

Now if we premultiply the constraint by B^{-1} we get $X_B = B^{-1}b - B^{-1}NX_N$. Eliminating X_B, the objective function becomes

$$C_B B^{-1} b - C_B B^{-1} N X_N + C_N X_N$$

We can rewrite this expression as

$$C_B B^{-1} b + (C_N - C_B B^{-1} N) X_N, \qquad \text{where}$$

$C_B B^{-1} b$ is the value of Z_o and $(C_N - C_B B^{-1} N)$ gives the coefficients of the non-basic variables (given in row zero). Since $C_B - C_B B^{-1} B = 0$, we can write the complete vector of row zero coefficients as $C - C_B B^{-1} A = C - \pi A$ where π is the dual vector.

Now consider the identity matrix obtained from the columns of the basic variables in the initial tableau. Let C_I be its coefficients in the objective function and let C_I' be its updated coefficients. Then

$$C_I' = C_I - \pi I \qquad \text{or} \qquad \pi = C_I - C_I' .$$

BIBLIOGRAPHY

1. Balinski, M. L. and Quandt, R. E., "On an Integer Program for a Delivery Problem," Operations Research, Vol. 12, pp. 300-304 (1964)

2. Bellmore, M., Bennington, G., and Lubore, S., "A Maximum Utility Solution to a Vehicle Constrained Tanker Scheduling Problem," Naval Research Logistics Quarterly, Vol. 15, pp. 403-411 (1968).

3. Bellmore, M., Liebman, J., and Marks, D., "An Extension of the (Szwarc) Truck Assignment Problem," Naval Research Logistics Quarterly, Vol. 19, pp. 91-99 (1972).

4. Braun, W., "A Computerized Simulation Approach to the Solution of the Carrier Dispatching Problem," Master's Thesis, Kansas State University (1967).

5. Christofides, N. and Eilon, S., "An Algorithm for the Vehicle Dispatching Problem," Operations Research Quarterly, Vol. 20, pp. 309-318 (1969).

6. Clarke, G. and Wright, J. W., "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," Operations Research, Vol. 12, pp. 568-581 (1964).

7. Conway, R. W., Maxwell, W. L., and Miller, L. W., Theory of Scheduling, Addison-Wesley Publishing Company, Reading, Massachusetts (1967).

8. Dantzig, G. B., "Application of the Simplex Method to a Transportation Problem," Cowles Commission Monograph, No. 13 (1951).

9. Dantzig, G. B. and Fulkerson, D. R., "Minimizing the Number of Tanker's to meet a Fixed Schedule," Naval Research Logistics Quarterly, Vol. 1, pp. 217-222 (1954).

10. Dantzig, G. B. and Ramser, H. J., "The Truck Dispatching Problem," Management Science, Vol. 6, pp. 80-91 (1959).

11. Eilon, S., Watson-Gandy, C. D. T., and Christofides, N., Distribution Management: Mathematical Modelling and Practical Analysis, Hafner Publishing Company, New York (1971).

12. Ford, L. R. and Fulkerson, D. R., Flows in Networks, Princeton University Press, Princeton, New Jersey (1962).

13. Gaskell, J. J., "Bases for Vehicle Fleet Scheduling," Operations Research Quarterly, Vol. 18, pp. 281-295 (1967).

14. Gavish, B. and Schweitzer, P., "An Algorithm for Combining Truck Trips," Operations Research, Statistics and Economics Mimeograph Series No. 105, Israel Institute of Technology, Haifa, Israel (1973)

15. Gomory, R. E., "An Algorithm for Integer Solutions to Linear Programs," in R. L. Graves and P. Wolfe, Recent Advances in Mathematical Programming, pp. 269-302, McGraw-Hill, New York (1963).

16. Gould, J., "The Size and Composition of a Road Transport Fleet," Operations Research Quarterly, Vol. 20, pp. 81-92 (1969).

17. Hausman, W. H. and Gilmour, P., "A Multiperiod Truck Delivery Problem," Transportation Research, Vol. 1, pp. 349-357 (1967).

18. Hayes, R. L., "The Delivery Problem," Carnegie Institute of Technology, Graduate School of Industrial Administration, Pittsburgh, Report No. MSR 106 (1967).

19. Held, M. and Karp, R. M., "A Dynamic Programming Approach to Sequencing Problems," Journal of the Society of Industrial and Applied Mathematics, Vol. 10, pp. 196-210 (1962).

20. Holladay, J., "Some Transportation Problems and Techniques for Solving Them," Naval Research Logistics Quarterly, Vol. 11, pp. 15-42 (1964).

21. Klein, M., "A Primal Method for Minimum Cost Flows with Applications to the Assignment and Transportation Problems," Management Science, Vol. 14, pp. 205-220 (1967).

22. Knight, K. W. and Hofer, J. P., "Vehicle Scheduling with Timed and Connected Calls; a Case Study," Operations Research Quarterly, Vol. 19, pp. 299-310 (1968).

23. Knowles, K., "The Use of a Heuristic Tree - Search Algorithm for Vehicle Routing and Scheduling," Operational Research Conference, Exeter, England (1967).

24. Lasdon, L. S., Optimization Theory for Large Systems, MacMillan, New York (1970).

25. Lawler, E. L. and Wood, D. E., "Branch-and-Bound Methods: a Survey," Operations Research, Vol. 14, pp. 699-719 (1966).

26. Little, J. D. C., et al., "An Algorithm for the Traveling Salesman Problem," Operations Research, Vol. 11, pp. 972-989 (1963).

27. Martin-Lof, A., "A Branch-and-Bound Algorithm for Determining the Minimal Fleet Size of a Transportation System," Transportation Science, Vol. 4, pp. 159-163 (1970).

28. McDonald, J. J., "Vehicle Scheduling - A Case Study," Operations Research Quarterly, Vol. 23, pp. 433-444 (1972).

29. National Computing Centere, Ltd., Computers in Vehicle Scheduling, National Computing Centere, Ltd., Manchester, England (1969).

30. Newton, R. M. and Thomas, W. H., "Design of School Bus Routes by Digital Computer," paper presented to the 13th National Meeting of the O. R. S. A.

31. Norman, W. E., The Impact of Computer Techniques on Road Transport Planning, National Computing Centere, Ltd., Manchester, England (1969).

32. Peters, S., "An Extension of the Transportation Problem," Zeitschrift Fur Operations Research," Vol. 16, pp. B123-143 (1972).

33. Pierce, J. F., "Direct Search Algorithms for Truck Dispatching Problems, Part 1," Transportation Research, Vol. 3, pp. 3-42 (1969).

34. Pullen, H. G. M. and Webb, M. H. J., "A Computer Application to a Transport Scheduling Problem," Computer Journal, Vol. 10, pp. 10-13 (1967).

35. Schrage, L., "Using Decomposition in Integer Programming," Naval Research Logistics Quarterly, Vol. 20, pp. 469-476 (1973).

36. Szwarc, W., "The Truck Assignment Problem," Naval Research Logistics Quarterly, Vol. 14, pp. 529-557 (1967).

37. Tillman, F. A., "The Multi-Terminal Delivery Problem with Probabalistic Demands," Transportation Science, Vol. 3, pp. 192-204 (1969).

38. Tillman, F. A. and Cochran, H., "A Heuristic Approach for Solving the Delivery Problem," Journal of Industrial Engineering, Vol. 19, pp. 354-358 (1968).

39. Tillman, F. A. and Hering, R. W., "A study of a Look-Ahead Procedure for Solving the Multi-Terminal Delivery Problem," Transportation Research, Vol. 5, pp. 225-229 (1971).

40. Trotter, L. E., "An Implicit Enumeration Algorithm for Integer Programming," Master's Thesis, Georgia Institute of Technology (1970).

41. Trotter, L. E., "User's Instruction for the Integer Programming Code ENUMER-8," Technical Report #1347, Mathematics Research Center, Unviersity of Wisconsin-Madison (1973).

42. Trotter, L. E. and Shetty, C. M., "An Algorithm for the Bounded Variable Integer Programming Problem," Technical Summary Report #1355, Mathematics Research Center, University of Wisconsin-Madison (1973).

43. Tyagi, M. S., "A Practical Method for Truck Dispatching Problem," Journal of the Operations Research Society of Japan, Vol. 10, pp. 76-92 (1968).

44. Univac, Linear Programming, Sperry Rand Corporation (1966).

45. Wagner, H. M., Principles of Operations Research (with Applications to Managerial Decisions), Prentice-Hall, Inc., New Jersey (1969).

46. Wren, A., Computers in Transport Planning and Operation, Ian Allen, London (1971).

47. Wren, A. and Holliday, A., "Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points," Operations Research Quarterly, Vol. 23, pp. 333-344 (1972).

48. Dijkstra, E. W., "A Note on Two Problem in Connexion with Graphs," Numerische Mathematik, Vol. 1, pp. 269-271 (1959).

49. Dantzig, G. B., Blattner, W. O., and Rao, M. R., "All Shortest Routes from a Fixed Origin in a Graph," Technical Report No. 66-2, Operations Research House, Stanford University (1966).

50. Gavish, B. and Schweitzer, P., "An Algorithm for Combining Truck Trips," Transportation Science, Vol. 8, pp. 13-23 (1974)